

UNIVERSAL ALGORITHMS, MATHEMATICS OF SEMIRINGS AND PARALLEL COMPUTATIONS

G. L. LITVINOV^{*1,2}, V. P. MASLOV³, A. YA. RODIONOV²,
AND A. N. SOBOLEVSKI^{1,4}

ABSTRACT. This is a survey paper on applications of mathematics of semirings to numerical analysis and computing. Concepts of universal algorithm and generic program are discussed. Relations between these concepts and mathematics of semirings are examined. A very brief introduction to mathematics of semirings (including idempotent and tropical mathematics) is presented. Concrete applications to optimization problems, idempotent linear algebra and interval analysis are indicated. It is known that some nonlinear problems (and especially optimization problems) become linear over appropriate semirings with idempotent addition (the so-called idempotent superposition principle). This linearity over semirings is convenient for parallel computations.

Key words and phrases: semirings, idempotent semirings, universal algorithms, generic programs, correspondence principle, superposition principle, optimization on graphs, linear algebra over semirings, interval analysis, parallel computations, hardware and software design.

2000 MSC: primary 15A80, 20M99, 65F99, 65K10, 68N19, 65G30, 68W10; secondary 68N30, 68Q65, 49M99, 12K10.

Contents

1. Introduction
2. Universal algorithms
3. Universal algorithms and accuracy of computations

To appear in *Springer Lecture Notes in Computational Science and Engineering*.

* Corresponding author: glitvinov@gmail.com.

¹ J. V. Poncelet Laboratory (UMI 2615 CNRS).

² D. V. Skobel'syn Research Institute for Nuclear Physics, Moscow State University.

³ Physics Faculty, Moscow State University.

⁴ A. A. Kharkevich Institute for Information Transmission Problems.

- 4. Mathematics of semirings
 - 4.1. Basic definitions
 - 4.2. Closure operations
 - 4.3. Matrices over semirings
 - 4.4. Discrete stationary Bellman equations
 - 4.5. Weighted directed graphs and matrices over semirings
- 5. Universal algorithms of linear algebra over semirings
- 6. The idempotent correspondence principle
- 7. The superposition principle and parallel computing
- 8. The correspondence principle for computations
- 9. The correspondence principle for hardware design
- 10. The correspondence principle for software design
- 11. Interval analysis in idempotent mathematics
- References

1. INTRODUCTION

It is well known that programmers are the laziest persons in the world. They constantly try to write less and get more. The so-called art of programming is just the way to achieve this goal. They started from programming in computer codes, found this practice boring and invented assembler, then macro assembler, then programming languages to reduce the amount of work (but not the salary). The next step was to separate algorithms and data. The new principle “Algorithm + data structure = program” was a great step in the same direction. Now some people could work on algorithms and express those in a more or less data structure independent manner; others programmers implement algorithms for different target data structures using different languages.

This scheme worked, and worked great, but had an important drawback: for the same algorithm one has to write a new program every time a new data type is required. Not only is this a boring process, it’s also consuming time and money and worse, it is error-prone. So the next step was to find how to write programs in a way independent of data structures. Object oriented languages, such as C++, Java and many others (see, e.g., [47, 62]) opened precisely such a way. For

C++, templates and STL give even more opportunities. This means that for a given algorithm one can vary a data structure while keeping the program unchanged. The principle was changed to “Algorithm + data structure family = program.”

But how does this approach work? What constitutes a “data structure family”? The answer is this: operations. Any algorithm manipulates data by means of a set of basic operations. For example for sorting the comparison operation is required; for scalar products and matrix multiplications, the operations of addition and multiplication are required. So to use a new data structure with the same “generic” program one has to implement the required operations for this data.

The hope was that some standard libraries of generic programs would cover almost all programmers’ needs and programming would be very much like Lego constructing. Well, to some extent it worked, but not as much as it was expected. Why? For different reasons. Some of them are economical. Why invest time and money in solving generic needs when a fast patch exists? Indeed, who would buy the next release if the current one is perfect? Another reason: there is not a great variety of possible data structures for most popular algorithms. In linear algebra one can use floating numbers, double precision numbers, infinite precision floating numbers, rational numbers, rational numbers with fix precision, Hensel codes, complex numbers, integers. Not much.

Mathematics of semirings gives a new approach to the generic programming. It parameterized algorithms. The new principle is “Parameterized algorithm + data structure family = program.” What are these parameters? They are operations (e.g., addition and multiplication). Sounds great, but does it really work? Yes. And we will show how and why. For example, we will show how the same old algorithm of linear algebra transformed by a simple redefinition of operations, can be applied to different problems in different areas. For example, the same program for solving systems of linear equations can be applied to the shortest path problem and other optimization problems (and interval versions of the problems) by means of simple redefinitions of the addition and multiplication operations. The algorithm was changed (by changing parameters, i.e. operations), the data structure

was changed, the nature of the problem was changed, but the program was not changed!

There are deep mathematical reasons (related to mathematics of semirings, idempotent and tropical mathematics) why this approach works. We will briefly discuss them in this paper.

The concept of a generic program was introduced by many authors; for example, in [36] such programs were called ‘program schemes.’ In this paper, we discuss *universal algorithms* implemented in the form of generic programs and their specific features. This paper is closely related to papers [38–40, 42, 43], in which the concept of a universal algorithm was defined and software and hardware implementation of such algorithms was discussed in connection with problems of idempotent mathematics [31, 32, 37–46, 48–53, 55, 71, 72]. In the present paper the emphasis is placed on software and hardware implementations of universal algorithms, computation with arbitrary accuracy, universal algorithms of linear algebra over semirings, and their implementations.

We also present a very brief introduction to mathematics of semirings and especially to the mathematics of *idempotent semirings* (i.e. semirings with idempotent addition). Mathematics over idempotent semirings is called *idempotent mathematics*. The so-called idempotent correspondence principle and idempotent superposition principle (see [38–43, 49–53] are discussed.

There exists a correspondence between interesting, useful, and important constructions and results concerning the field of real (or complex) numbers and similar constructions dealing with various idempotent semirings. This correspondence can be formulated in the spirit of the well-known N. Bohr’s *correspondence principle* in quantum mechanics; in fact, the two principles are intimately connected (see [38–40]). In a sense, the traditional mathematics over numerical fields can be treated as a ‘quantum’ theory, whereas the idempotent mathematics can be treated as a ‘classical’ shadow (or counterpart) of the traditional one. It is important that the idempotent correspondence principle is valid for algorithms, computer programs and hardware units.

In quantum mechanics the *superposition principle* means that the Schrödinger equation (which is basic for the theory) is linear. Similarly in idempotent mathematics the (idempotent) superposition principle (formulated by V. P. Maslov) means that some important and basic problems and equations that are nonlinear in the usual sense (e.g., the Hamilton-Jacobi equation, which is basic for classical mechanics and appears in many optimization problems, or the Bellman equation and its versions and generalizations) can be treated as linear over appropriate idempotent semirings, see [38–41, 48–51].

Note that numerical algorithms for infinite-dimensional linear problems over idempotent semirings (e.g., idempotent integration, integral operators and transformations, the Hamilton–Jacobi and generalized Bellman equations) deal with the corresponding finite-dimensional approximations. Thus idempotent linear algebra is the basis of the idempotent numerical analysis and, in particular, the *discrete optimization theory*.

B. A. Carré [10, 11] (see also [7, 23–25]) used the idempotent linear algebra to show that different optimization problems for finite graphs can be formulated in a unified manner and reduced to solving Bellman equations, i.e., systems of linear algebraic equations over idempotent semirings. He also generalized principal algorithms of computational linear algebra to the idempotent case and showed that some of these coincide with algorithms independently developed for solution of optimization problems; for example, Bellman’s method of solving the shortest path problem corresponds to a version of Jacobi’s method for solving a system of linear equations, whereas Ford’s algorithm corresponds to a version of Gauss–Seidel’s method. We briefly discuss Bellman equations and the corresponding optimization problems on graphs.

We stress that these well-known results can be interpreted as a manifestation of the idempotent superposition principle.

We also briefly discuss interval analysis over idempotent and positive semirings. Idempotent interval analysis appears in [45, 46, 69]. Later many authors dealt with this subject, see, e.g. [12, 20, 27, 57, 58, 77].

It is important to observe that intervals over an idempotent semiring form a new idempotent semiring. Hence universal algorithms can be applied to elements of this new semiring and generate interval versions of the initial algorithms.

Note finally that idempotent mathematics is remarkably simpler than its traditional analog.

2. UNIVERSAL ALGORITHMS

Computational algorithms are constructed on the basis of certain primitive operations. These operations manipulate data that describe “numbers.” These “numbers” are elements of a “numerical domain,” i.e., a mathematical object such as the field of real numbers, the ring of integers, or an idempotent semiring of numbers (idempotent semirings and their role in idempotent mathematics are discussed in [10, 11, 22–26, 31, 32, 37–43] and below in this paper).

In practice elements of the numerical domains are replaced by their computer representations, i.e., by elements of certain finite models of these domains. Examples of models that can be conveniently used for computer representation of real numbers are provided by various modifications of floating point arithmetics, approximate arithmetics of rational numbers [44], and interval arithmetics. The difference between mathematical objects (“ideal” numbers) and their finite models (computer representations) results in computational (e.g., rounding) errors.

An algorithm is called *universal* if it is independent of a particular numerical domain and/or its computer representation. A typical example of a universal algorithm is the computation of the scalar product (x, y) of two vectors $x = (x_1, \dots, x_n)$ and $y = (y_1, \dots, y_n)$ by the formula $(x, y) = x_1y_1 + \dots + x_ny_n$. This algorithm (formula) is independent of a particular domain and its computer implementation, since the formula is well-defined for any semiring. It is clear that one algorithm can be more universal than another. For example, the simplest Newton–Cotes formula, the rectangular rule, provides the most universal algorithm for numerical integration; indeed, this formula is valid even for idempotent integration (over any idempotent semiring, see below and [4, 32, 38–43, 48–51]. Other quadrature formulas (e.g.,

combined trapezoid rule or the Simpson formula) are independent of computer arithmetics and can be used (e.g., in the iterative form) for computations with arbitrary accuracy. In contrast, algorithms based on Gauss–Jacobi formulas are designed for fixed accuracy computations: they include constants (coefficients and nodes of these formulas) defined with fixed accuracy. Certainly, algorithms of this type can be made more universal by including procedures for computing the constants; however, this results in an unjustified complication of the algorithms.

Computer algebra algorithms used in such systems as Mathematica, Maple, REDUCE, and others are highly universal. Most of the standard algorithms used in linear algebra can be rewritten in such a way that they will be valid over any field and complete idempotent semiring (including semirings of intervals; see below and [45, 46, 69], where an interval version of the idempotent linear algebra and the corresponding universal algorithms are discussed).

As a rule, iterative algorithms (beginning with the successive approximation method) for solving differential equations (e.g., methods of Euler, Euler–Cauchy, Runge–Kutta, Adams, a number of important versions of the difference approximation method, and the like), methods for calculating elementary and some special functions based on the expansion in Taylor’s series and continuous fractions (Padé approximations) and others are independent of the computer representation of numbers.

3. UNIVERSAL ALGORITHMS AND ACCURACY OF COMPUTATIONS

Calculations on computers usually are based on a floating-point arithmetic with a mantissa of a fixed length; i.e., computations are performed with fixed accuracy. Broadly speaking, with this approach only the relative rounding error is fixed, which can lead to a drastic loss of accuracy and invalid results (e.g., when summing series and subtracting close numbers). On the other hand, this approach provides rather high speed of computations. Many important numerical algorithms are designed to use floating-point arithmetic (with fixed accuracy) and ensure the maximum computation speed. However, these algorithms

are not universal. The above mentioned Gauss–Jacobi quadrature formulas, computation of elementary and special functions on the basis of the best polynomial or rational approximations or Padé–Chebyshev approximations, and some others belong to this type. Such algorithms use nontrivial constants specified with fixed accuracy.

Recently, problems of accuracy, reliability, and authenticity of computations (including the effect of rounding errors) have gained much attention; in part, this fact is related to the ever-increasing performance of computer hardware. When errors in initial data and rounding errors strongly affect the computation results, such as in ill-posed problems, analysis of stability of solutions, etc., it is often useful to perform computations with improved and variable accuracy. In particular, the rational arithmetic, in which the rounding error is specified by the user [44], can be used for this purpose. This arithmetic is a useful complement to the interval analysis [54]. The corresponding computational algorithms must be universal (in the sense that they must be independent of the computer representation of numbers).

4. MATHEMATICS OF SEMIRINGS

A broad class of universal algorithms is related to the concept of a semiring. We recall here the definition of a semiring (see, e.g., [21, 22]).

4.1. Basic definitions. Consider a semiring, i.e., a set S endowed with two associative operations: *addition* \oplus and *multiplication* \odot such that addition is commutative, multiplication distributes over addition from either side, $\mathbf{0}$ (resp., $\mathbf{1}$) is the neutral element of addition (resp., multiplication), $\mathbf{0} \odot x = x \odot \mathbf{0} = \mathbf{0}$ for all $x \in S$, and $\mathbf{0} \neq \mathbf{1}$. Let the semiring S be partially ordered by a relation \preccurlyeq such that $\mathbf{0}$ is the least element and the inequality $x \preccurlyeq y$ implies that $x \oplus z \preccurlyeq y \oplus z$, $x \odot z \preccurlyeq y \odot z$, and $z \odot x \preccurlyeq z \odot y$ for all $x, y, z \in S$; in this case the semiring S is called *positive* (see, e.g., [22]).

A semiring S is called a *semifield* if every nonzero element is invertible.

A semiring S is called *idempotent* if $x \oplus x = x$ for all $x \in S$, see, e.g., [4, 5, 7, 9, 10, 21–24, 31, 37–43, 48–53]. In this case the addition \oplus

defines a *canonical partial order* \preceq on the semiring S by the rule: $x \preceq y$ iff $x \oplus y = y$. It is easy to prove that any idempotent semiring is positive with respect to this order. Note also that $x \oplus y = \sup\{x, y\}$ with respect to the canonical order. In the sequel, we shall assume that all idempotent semirings are ordered by the canonical partial order relation.

We shall say that a positive (e.g., idempotent) semiring S is *complete* if it is complete as an ordered set. This means that for every subset $T \subset S$ there exist elements $\sup T \in S$ and $\inf T \in S$.

The most well-known and important examples of positive semirings are “numerical” semirings consisting of (a subset of) real numbers and ordered by the usual linear order \leq on \mathbf{R} : the semiring \mathbf{R}_+ with the usual operations $\oplus = +$, $\odot = \cdot$ and neutral elements $\mathbf{0} = 0$, $\mathbf{1} = 1$, the semiring $\mathbf{R}_{\max} = \mathbf{R} \cup \{-\infty\}$ with the operations $\oplus = \max$, $\odot = +$ and neutral elements $\mathbf{0} = -\infty$, $\mathbf{1} = 0$, the semiring $\widehat{\mathbf{R}}_{\max} = \mathbf{R}_{\max} \cup \{\infty\}$, where $x \preceq \infty$, $x \oplus \infty = \infty$ for all x , $x \odot \infty = \infty \odot x = \infty$ if $x \neq \mathbf{0}$, and $\mathbf{0} \odot \infty = \infty \odot \mathbf{0}$, and the semiring $S_{\max, \min}^{[a, b]} = [a, b]$, where $-\infty \leq a < b \leq +\infty$, with the operations $\oplus = \max$, $\odot = \min$ and neutral elements $\mathbf{0} = a$, $\mathbf{1} = b$. The semirings \mathbf{R}_{\max} , $\widehat{\mathbf{R}}_{\max}$, and $S_{\max, \min}^{[a, b]} = [a, b]$ are idempotent. The semirings $\widehat{\mathbf{R}}_{\max}$, $S_{\max, \min}^{[a, b]}$, $\widehat{\mathbf{R}}_+ = \mathbf{R}_+ \cup \{\infty\}$ are complete. Remind that every partially ordered set can be imbedded to its completion (a minimal complete set containing the initial one).

The semiring $\mathbf{R}_{\min} = \mathbf{R} \cup \{\infty\}$ with operations $\oplus = \min$ and $\odot = +$ and neutral elements $\mathbf{0} = \infty$, $\mathbf{1} = 0$ is isomorphic to \mathbf{R}_{\max} .

The semiring \mathbf{R}_{\max} is also called the *max-plus algebra*. The semifields \mathbf{R}_{\max} and \mathbf{R}_{\min} are called *tropical algebras*. The term “tropical” initially appeared in [68] for a discrete version of the max-plus algebra as a suggestion of Ch. Choffrut, see also [26, 55, 61, 72].

Many mathematical constructions, notions, and results over the fields of real and complex numbers have nontrivial analogs over idempotent semirings. Idempotent semirings have become recently the object of investigation of new branches of mathematics, *idempotent mathematics* and *tropical geometry*, see, e.g. [5, 13, 15–19, 35–28, 31, 32, 37–46, 48–53, 55, 71, 72].

4.2. Closure operations. Let a positive semiring S be endowed with a partial unary *closure operation* $*$ such that $x \preccurlyeq y$ implies $x^* \preccurlyeq y^*$ and $x^* = \mathbf{1} \oplus (x^* \odot x) = \mathbf{1} \oplus (x \odot x^*)$ on its domain of definition. In particular, $\mathbf{0}^* = \mathbf{1}$ by definition. These axioms imply that $x^* = \mathbf{1} \oplus x \oplus x^2 \oplus \cdots \oplus (x^* \odot x^n)$ if $n \geq 1$. Thus x^* can be considered as a ‘regularized sum’ of the series $x^* = \mathbf{1} \oplus x \oplus x^2 \oplus \cdots$; in an idempotent semiring, by definition, $x^* = \sup\{\mathbf{1}, x, x^2, \dots\}$ if this supremum exists. So if S is complete, then the closure operation is well-defined for every element $x \in S$.

In numerical semirings the operation $*$ is defined as follows: $x^* = (1 - x)^{-1}$ if $x \prec 1$ in \mathbf{R}_+ , or $\widehat{\mathbf{R}}_+$ and $x^* = \infty$ if $x \succ 1$ in $\widehat{\mathbf{R}}_+$; $x^* = \mathbf{1}$ if $x \preccurlyeq \mathbf{1}$ in \mathbf{R}_{\max} and $\widehat{\mathbf{R}}_{\max}$, $x^* = \infty$ if $x \succ \mathbf{1}$ in $\widehat{\mathbf{R}}_{\max}$, $x^* = \mathbf{1}$ for all x in $S_{\max, \min}^{[a, b]}$. In all other cases x^* is undefined. Note that the closure operation is very easy to implement.

4.3. Matrices over semirings. Denote by $\text{Mat}_{mn}(S)$ a set of all matrices $A = (a_{ij})$ with m rows and n columns whose coefficients belong to a semiring S . The sum $A \oplus B$ of matrices $A, B \in \text{Mat}_{mn}(S)$ and the product AB of matrices $A \in \text{Mat}_{lm}(S)$ and $B \in \text{Mat}_{mn}(S)$ are defined according to the usual rules of linear algebra: $A \oplus B = (a_{ij} \oplus b_{ij}) \in \text{Mat}_{mn}(S)$ and

$$AB = \left(\bigoplus_{k=1}^m a_{ik} \odot b_{kj} \right) \in \text{Mat}_{ln}(S),$$

where $A \in \text{Mat}_{lm}(S)$ and $B \in \text{Mat}_{mn}(S)$. Note that we write AB instead of $A \odot B$.

If the semiring S is positive, then the set $\text{Mat}_{mn}(S)$ is ordered by the relation $A = (a_{ij}) \preccurlyeq B = (b_{ij})$ iff $a_{ij} \preccurlyeq b_{ij}$ in S for all $1 \leq i \leq m$, $1 \leq j \leq n$.

The matrix multiplication is consistent with the order \preccurlyeq in the following sense: if $A, A' \in \text{Mat}_{lm}(S)$, $B, B' \in \text{Mat}_{mn}(S)$ and $A \preccurlyeq A'$, $B \preccurlyeq B'$, then $AB \preccurlyeq A'B'$ in $\text{Mat}_{ln}(S)$. The set $\text{Mat}_{nn}(S)$ of square $(n \times n)$ matrices over a [positive, idempotent] semiring S forms a [positive, idempotent] semiring with a zero element $O = (o_{ij})$, where $o_{ij} = \mathbf{0}$, $1 \leq i, j \leq n$, and a unit element $E = (\delta_{ij})$, where $\delta_{ij} = \mathbf{1}$ if $i = j$ and $\delta_{ij} = \mathbf{0}$ otherwise.

The set Mat_{nn} is an example of a noncommutative semiring if $n > 1$.

The closure operation in matrix semirings over a positive semiring S can be defined inductively (another way to do that see in [22] and below): $A^* = (a_{11})^* = (a_{11}^*)$ in $\text{Mat}_{11}(S)$ and for any integer $n > 1$ and any matrix

$$A = \begin{pmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{pmatrix},$$

where $A_{11} \in \text{Mat}_{kk}(S)$, $A_{12} \in \text{Mat}_{k, n-k}(S)$, $A_{21} \in \text{Mat}_{n-k, k}(S)$, $A_{22} \in \text{Mat}_{n-k, n-k}(S)$, $1 \leq k \leq n$, by definition,

$$(1) \quad A^* = \begin{pmatrix} A_{11}^* \oplus A_{11}^* A_{12} D^* A_{21} A_{11}^* & A_{11}^* A_{12} D^* \\ D^* A_{21} A_{11}^* & D^* \end{pmatrix},$$

where $D = A_{22} \oplus A_{21} A_{11}^* A_{12}$. It can be proved that this definition of A^* implies that the equality $A^* = A^* A \oplus E$ is satisfied and thus A^* is a ‘regularized sum’ of the series $E \oplus A \oplus A^2 \oplus \dots$.

Note that this recurrence relation coincides with the formulas of escalator method of matrix inversion in the traditional linear algebra over the field of real or complex numbers, up to the algebraic operations used. Hence this algorithm of matrix closure requires a polynomial number of operations in n .

4.4. Discrete stationary Bellman equations. Let S be a positive semiring. The *discrete stationary Bellman equation* has the form

$$(2) \quad X = AX \oplus B,$$

where $A \in \text{Mat}_{nn}(S)$, $X, B \in \text{Mat}_{ns}(S)$, and the matrix X is unknown. Let A^* be the closure of the matrix A . It follows from the identity $A^* = A^* A \oplus E$ that the matrix $A^* B$ satisfies this equation; moreover, it can be proved that for positive semirings this solution is the least in the set of solutions to equation (2) with respect to the partial order in $\text{Mat}_{ns}(S)$.

4.5. Weighted directed graphs and matrices over semirings. Suppose that S is a semiring with zero $\mathbf{0}$ and unity $\mathbf{1}$. It is well-known that any square matrix $A = (a_{ij}) \in \text{Mat}_{nn}(S)$ specifies a *weighted directed graph*. This geometrical construction includes three kinds of

objects: the set X of n elements x_1, \dots, x_n called *nodes*, the set Γ of all ordered pairs (x_i, x_j) such that $a_{ij} \neq \mathbf{0}$ called *arcs*, and the mapping $A: \Gamma \rightarrow S$ such that $A(x_i, x_j) = a_{ij}$. The elements a_{ij} of the semiring S are called *weights* of the arcs.

Conversely, any given weighted directed graph with n nodes specifies a unique matrix $A \in \text{Mat}_{nn}(S)$.

This definition allows for some pairs of nodes to be disconnected if the corresponding element of the matrix A is $\mathbf{0}$ and for some channels to be “loops” with coincident ends if the matrix A has nonzero diagonal elements. This concept is convenient for analysis of parallel and distributed computations and design of computing media and networks (see, e.g., [4, 53, 73]).

Recall that a sequence of nodes of the form

$$p = (y_0, y_1, \dots, y_k)$$

with $k \geq 0$ and $(y_i, y_{i+1}) \in \Gamma$, $i = 0, \dots, k-1$, is called a *path* of length k connecting y_0 with y_k . Denote the set of all such paths by $P_k(y_0, y_k)$. The weight $A(p)$ of a path $p \in P_k(y_0, y_k)$ is defined to be the product of weights of arcs connecting consecutive nodes of the path:

$$A(p) = A(y_0, y_1) \odot \dots \odot A(y_{k-1}, y_k).$$

By definition, for a ‘path’ $p \in P_0(x_i, x_j)$ of length $k = 0$ the weight is $\mathbf{1}$ if $i = j$ and $\mathbf{0}$ otherwise.

For each matrix $A \in \text{Mat}_{nn}(S)$ define $A^0 = E = (\delta_{ij})$ (where $\delta_{ij} = \mathbf{1}$ if $i = j$ and $\delta_{ij} = \mathbf{0}$ otherwise) and $A^k = AA^{k-1}$, $k \geq 1$. Let $a_{ij}^{(k)}$ be the (i, j) th element of the matrix A^k . It is easily checked that

$$a_{ij}^{(k)} = \bigoplus_{\substack{i_0=i, i_k=j \\ 1 \leq i_1, \dots, i_{k-1} \leq n}} a_{i_0 i_1} \odot \dots \odot a_{i_{k-1} i_k}.$$

Thus $a_{ij}^{(k)}$ is the supremum of the set of weights corresponding to all paths of length k connecting the node $x_{i_0} = x_i$ with $x_{i_k} = x_j$.

Denote the elements of the matrix A^* by $a_{ij}^{(*)}$, $i, j = 1, \dots, n$; then

$$a_{ij}^{(*)} = \bigoplus_{0 \leq k < \infty} \bigoplus_{p \in P_k(x_i, x_j)} A(p).$$

The closure matrix A^* solves the well-known *algebraic path problem*, which is formulated as follows: for each pair (x_i, x_j) calculate the supremum of weights of all paths (of arbitrary length) connecting node x_i with node x_j . The closure operation in matrix semirings has been studied extensively (see, e.g., [1, 2, 5–7, 10, 11, 15–17, 22–26, 31, 32, 46] and references therein).

Example 1. The shortest path problem. Let $S = \mathbf{R}_{\min}$, so the weights are real numbers. In this case

$$A(p) = A(y_0, y_1) + A(y_1, y_2) + \cdots + A(y_{k-1}, y_k).$$

If the element a_{ij} specifies the length of the arc (x_i, x_j) in some metric, then $a_{ij}^{(*)}$ is the length of the shortest path connecting x_i with x_j .

Example 2. The maximal path width problem. Let $S = \mathbf{R} \cup \{0, 1\}$ with $\oplus = \max$, $\odot = \min$. Then

$$a_{ij}^{(*)} = \max_{p \in \bigcup_{k \geq 1} P_k(x_i, x_j)} A(p), \quad A(p) = \min(A(y_0, y_1), \dots, A(y_{k-1}, y_k)).$$

If the element a_{ij} specifies the “width” of the arc (x_i, x_j) , then the width of a path p is defined as the minimal width of its constituting arcs and the element $a_{ij}^{(*)}$ gives the supremum of possible widths of all paths connecting x_i with x_j .

Example 3. A simple dynamic programming problem. Let $S = \mathbf{R}_{\max}$ and suppose a_{ij} gives the *profit* corresponding to the transition from x_i to x_j . Define the vector $B = (b_i) \in \text{Mat}_{n1}(\mathbf{R}_{\max})$ whose element b_i gives the *terminal profit* corresponding to exiting from the graph through the node x_i . Of course, negative profits (or, rather, losses) are allowed. Let m be the total profit corresponding to a path $p \in P_k(x_i, x_j)$, i.e.

$$m = A(p) + b_j.$$

Then it is easy to check that the supremum of profits that can be achieved on paths of length k beginning at the node x_i is equal to $(A^k B)_i$ and the supremum of profits achievable without a restriction on the length of a path equals $(A^* B)_i$.

Example 4. The matrix inversion problem. Note that in the formulas of this section we are using distributivity of the multiplication \odot with respect to the addition \oplus but do not use the idempotency axiom.

Thus the algebraic path problem can be posed for a nonidempotent semiring S as well (see, e.g., [66]). For instance, if $S = \mathbf{R}$, then

$$A^* = E + A + A^2 + \cdots = (E - A)^{-1}.$$

If $\|A\| > 1$ but the matrix $E - A$ is invertible, then this expression defines a regularized sum of the divergent matrix power series $\sum_{i \geq 0} A^i$.

There are many other important examples of problems (in different areas) related to algorithms of linear algebra over semirings (transitive closures of relations, accessible sets, critical paths, paths of greatest capacities, the most reliable paths, interval and other problems), see [1, 2, 4, 5, 10–13, 15–18, 20, 22–27, 31, 32, 45, 46, 53, 57, 58, 63–68, 74–77].

We emphasize that this connection between the matrix closure operation and solution to the Bellman equation gives rise to a number of different algorithms for numerical calculation of the closure matrix. All these algorithms are adaptations of the well-known algorithms of the traditional computational linear algebra, such as the Gauss–Jordan elimination, various iterative and escalator schemes, etc. This is a special case of the idempotent superposition principle (see below).

In fact, the theory of the discrete stationary Bellman equation can be developed using the identity $A^* = AA^* \oplus E$ as an additional axiom without any substantive interpretation (the so-called *closed semirings*, see, e.g., [7, 22, 36, 66]).

5. UNIVERSAL ALGORITHMS OF LINEAR ALGEBRA OVER SEMIRINGS

The most important linear algebra problem is to solve the system of linear equations

$$(3) \quad AX = B,$$

where A is a matrix with elements from the basic field and X and B are vectors (or matrices) with elements from the same field. It is required to find X if A and B are given. If A in (3) is not the identity matrix I , then system (3) can be written in form (2), i.e.,

$$X = AX + B. \quad (2')$$

It is well known that the form (2) or (2') is convenient for using the successive approximation method. Applying this method with the initial approximation $X_0 = 0$, we obtain the solution

$$(4) \quad X = A^*B,$$

where

$$(5) \quad A^* = I + A + A^2 + \cdots + A^n + \cdots$$

On the other hand, it is clear that

$$(6) \quad A^* = (I - A)^{-1},$$

if the matrix $I - A$ is invertible. The inverse matrix $(I - A)^{-1}$ can be considered as a regularized sum of the formal series (5).

The above considerations can be extended to a broad class of semirings.

The closure operation for matrix semirings $\text{Mat}_n(S)$ can be defined and computed in terms of the closure operation for S (see section 4.3 above); some methods are described in [1, 2, 7, 10, 11, 22, 23–25, 32, 35, 42, 46, 65–67]. One such method is described below (*LDM-factorization*).

If S is a field, then, by definition, $x^* = (1 - x)^{-1}$ for any $x \neq 1$. If S is an idempotent semiring, then, by definition,

$$(7) \quad x^* = \mathbf{1} \oplus x \oplus x^2 \oplus \cdots = \sup\{\mathbf{1}, x, x^2, \dots\},$$

if this supremum exists. Recall that it exists if S is complete, see section 4.2.

Consider a nontrivial universal algorithm applicable to matrices over semirings with the closure operation defined.

Example 5: Semiring LDM-Factorization.

Factorization of a matrix into the product $A = LDM$, where L and M are lower and upper triangular matrices with a unit diagonal, respectively, and D is a diagonal matrix, is used for solving matrix equations $AX = B$. We construct a similar decomposition for the Bellman equation $X = AX \oplus B$.

For the case $AX = B$, the decomposition $A = LDM$ induces the following decomposition of the initial equation:

$$(8) \quad LZ = B, \quad DY = Z, \quad MX = Y.$$

Hence, we have

$$(9) \quad A^{-1} = M^{-1}D^{-1}L^{-1},$$

if A is invertible. In essence, it is sufficient to find the matrices L , D and M , since the linear system (8) is easily solved by a combination of the forward substitution for Z , the trivial inversion of a diagonal matrix for Y , and the back substitution for X .

Using (8) as a pattern, we can write

$$(10) \quad Z = LZ \oplus B, \quad Y = DY \oplus Z, \quad X = MX \oplus Y.$$

Then

$$(11) \quad A^* = M^*D^*L^*.$$

A triple (L, D, M) consisting of a lower triangular, diagonal, and upper triangular matrices is called an *LDM-factorization* of a matrix A if relations (10) and (11) are satisfied. We note that in this case, the principal diagonals of L and M are zero.

The modification of the notion of *LDM-factorization* used in matrix analysis for the equation $AX = B$ is constructed in analogy with the construct suggested by Carré in [10, 11] for *LU-factorization*.

We stress that the algorithm described below can be applied to matrix computations over any semiring under the condition that the unary operation $a \mapsto a^*$ is applicable every time it is encountered in the computational process. Indeed, when constructing the algorithm, we use only the basic semiring operations of addition \oplus and multiplication \odot and the properties of associativity, commutativity of addition, and distributivity of multiplication over addition.

If A is a symmetric matrix over a semiring with a commutative multiplication, the amount of computations can be halved, since M and L are mapped into each other under transposition.

We begin with the case of a triangular matrix $A = L$ (or $A = M$). Then, finding X is reduced to the forward (or back) substitution.

Forward substitution

We are given:

- $L = \|l_j^i\|_{i,j=1}^n$, where $l_j^i = \mathbf{0}$ for $i \leq j$ (a lower triangular matrix with a zero diagonal);
- $B = \|b^i\|_{i=1}^n$.

It is required to find the solution $X = \|x^i\|_{i=1}^n$ to the equation $X = LX \oplus B$. The program fragment solving this problem is as follows.

for $i = 1$ to n do

{ $x^i := b^i$;
 for $j = 1$ to $i - 1$ do
 $x^i := x^i \oplus (l_j^i \odot x^j)$; }

Back substitution

We are given

- $M = \|m_j^i\|_{i,j=1}^n$, where $m_j^i = \mathbf{0}$ for $i \geq j$ (an upper triangular matrix with a zero diagonal);
- $B = \|b^i\|_{i=1}^n$.

It is required to find the solution $X = \|x^i\|_{i=1}^n$ to the equation $X = MX \oplus B$. The program fragment solving this problem is as follows.

for $i = n$ to 1 step -1 do

{ $x^i := b^i$;
 for $j = n$ to $i + 1$ step -1 do
 $x^i := x^i \oplus (m_j^i \odot x^j)$; }

Both algorithms require $(n^2 - n)/2$ operations \oplus and \odot .

Closure of a diagonal matrix

We are given

- $D = \text{diag}(d_1, \dots, d_n)$;
- $B = \|b^i\|_{i=1}^n$.

It is required to find the solution $X = \|x^i\|_{i=1}^n$ to the equation $X = DX \oplus B$. The program fragment solving this problem is as follows.

for $i = 1$ to n do

$x^i := (d_i)^* \odot b^i$;

This algorithm requires n operations $*$ and n multiplications \odot .

General case

We are given

- $L = \|l_j^i\|_{i,j=1}^n$, where $l_j^i = \mathbf{0}$ if $i \leq j$;
- $D = \text{diag}(d_1, \dots, d_n)$;
- $M = \|m_j^i\|_{i,j=1}^n$, where $m_j^i = \mathbf{0}$ if $i \geq j$;
- $B = \|b^i\|_{i=1}^n$.

It is required to find the solution $X = \|x^i\|_{i=1}^n$ to the equation $X = AX \oplus B$, where L , D , and M form the LDM -factorization of A . The program fragment solving this problem is as follows.

FORWARD SUBSTITUTION

for $i = 1$ to n do

{ $x^i := b^i$;

for $j = 1$ to $i - 1$ do

$x^i := x^i \oplus (l_j^i \odot x^j)$; }

CLOSURE OF A DIAGONAL MATRIX

for $i = 1$ to n do

$x^i := (d_i)^* \odot b^i$;

BACK SUBSTITUTION

for $i = n$ to 1 step -1 do

{ for $j = n$ to $i + 1$ step -1 do

$x^i := x^i \oplus (m_j^i \odot x^j)$; }

Note that x^i is not initialized in the course of the back substitution. The algorithm requires $n^2 - n$ operations \oplus , n^2 operations \odot , and n operations $*$.

LDM-factorization

We are given

- $A = \|a_j^i\|_{i,j=1}^n$.

It is required to find the LDM -factorization of A : $L = \|l_j^i\|_{i,j=1}^n$, $D = \text{diag}(d_1, \dots, d_n)$, and $M = \|m_j^i\|_{i,j=1}^n$, where $l_j^i = \mathbf{0}$ if $i \leq j$, and $m_j^i = \mathbf{0}$ if $i \geq j$.

The program uses the following internal variables:

- $C = \|c_j^i\|_{i,j=1}^n$;
- $V = \|v^i\|_{i=1}^n$;
- d .

INITIALISATION

for $i = 1$ to n do

 for $j = 1$ to n do

$$c_j^i = a_j^i;$$

MAIN LOOP

for $j = 1$ to n do

{ for $i = 1$ to j do

$$v^i := a_j^i;$$

 for $k = 1$ to $j - 1$ do

 for $i = k + 1$ to j do

$$v^i := v^i \oplus (a_k^i \odot v^k);$$

 for $i = 1$ to $j - 1$ do

$$a_j^i := (a_i^i)^* \odot v^i;$$

$$a_j^j := v^j;$$

 for $k = 1$ to $j - 1$ do

 for $i = j + 1$ to n do

$$a_j^i := a_j^i \oplus (a_k^i \odot v^k);$$

$$d = (v^j)^*;$$

 for $i = j + 1$ to n do

$$a_j^i := a_j^i \odot d; \}$$

This algorithm requires $(2n^3 - 3n^2 + n)/6$ operations \oplus , $(2n^3 + 3n^2 - 5n)/6$ operations \odot , and $n(n+1)/2$ operations $*$. After its completion, the matrices L , D , and M are contained, respectively, in the lower triangle, on the diagonal, and in the upper triangle of the matrix C . In the case when A is symmetric about the principal diagonal and the semiring over which the matrix is defined is commutative, the algorithm can be modified in such a way that the number of operations is reduced approximately by a factor of two.

Other examples can be found in [10, 11, 22–25, 35, 36, 66, 67].

Note that to compute the matrices A^* and A^*B it is convenient to solve the Bellman equation (2).

Some other interesting and important problems of linear algebra over semirings are examined, e.g., in [8, 9, 12, 18, 20, 22–25, 27, 57, 58, 59, 60, 74–77].

6. THE IDEMPOTENT CORRESPONDENCE PRINCIPLE

There is a nontrivial analogy between mathematics of semirings and quantum mechanics. For example, the field of real numbers can be treated as a “quantum object” with respect to idempotent semirings. So idempotent semirings can be treated as “classical” or “semi-classical” objects with respect to the field of real numbers.

Let \mathbf{R} be the field of real numbers and \mathbf{R}_+ the subset of all non-negative numbers. Consider the following change of variables:

$$u \mapsto w = h \ln u,$$

where $u \in \mathbf{R}_+ \setminus \{0\}$, $h > 0$; thus $u = e^{w/h}$, $w \in \mathbf{R}$. Denote by $\mathbf{0}$ the additional element $-\infty$ and by S the extended real line $\mathbf{R} \cup \{\mathbf{0}\}$. The above change of variables has a natural extension D_h to the whole S by $D_h(0) = \mathbf{0}$; also, we denote $D_h(1) = 0 = \mathbf{1}$.

Denote by S_h the set S equipped with the two operations \oplus_h (generalized addition) and \odot_h (generalized multiplication) such that D_h is a homomorphism of $\{\mathbf{R}_+, +, \cdot\}$ to $\{S, \oplus_h, \odot_h\}$. This means that $D_h(u_1 + u_2) = D_h(u_1) \oplus_h D_h(u_2)$ and $D_h(u_1 \cdot u_2) = D_h(u_1) \odot_h D_h(u_2)$, i.e., $w_1 \odot_h w_2 = w_1 + w_2$ and $w_1 \oplus_h w_2 = h \ln(e^{w_1/h} + e^{w_2/h})$. It is easy to prove that $w_1 \oplus_h w_2 \rightarrow \max\{w_1, w_2\}$ as $h \rightarrow 0$.

Denote by \mathbf{R}_{\max} the set $S = \mathbf{R} \cup \{\mathbf{0}\}$ equipped with operations $\oplus = \max$ and $\odot = +$, where $\mathbf{0} = -\infty$, $\mathbf{1} = 0$ as above. Algebraic structures in \mathbf{R}_+ and S_h are isomorphic; therefore \mathbf{R}_{\max} is a result of a deformation of the structure in \mathbf{R}_+ .

We stress the obvious analogy with the quantization procedure, where h is the analog of the Planck constant. In these terms, \mathbf{R}_+ (or \mathbf{R}) plays the part of a “quantum object” while \mathbf{R}_{\max} acts as a “classical” or “semi-classical” object that arises as the result of a *dequantization* of this quantum object.

Likewise, denote by \mathbf{R}_{\min} the set $\mathbf{R} \cup \{\mathbf{0}\}$ equipped with operations $\oplus = \min$ and $\odot = +$, where $\mathbf{0} = +\infty$ and $\mathbf{1} = 0$. Clearly, the corresponding dequantization procedure is generated by the change of variables $u \mapsto w = -h \ln u$.

Consider also the set $\mathbf{R} \cup \{\mathbf{0}, \mathbf{1}\}$, where $\mathbf{0} = -\infty$, $\mathbf{1} = +\infty$, together with the operations $\oplus = \max$ and $\odot = \min$. Obviously, it can be obtained as a result of a “second dequantization” of \mathbf{R} or \mathbf{R}_+ .

There is a natural transition from the field of real numbers or complex numbers to the semiring \mathbf{R}_{\max} (or \mathbf{R}_{\min}). This is a composition of the mapping $x \mapsto |x|$ and the deformation described above.

In general an *idempotent dequantization* is a transition from a basic field to an idempotent semiring in mathematical concepts, constructions and results, see [38–40] for details.

For example, the basic object of the traditional calculus is a *function* defined on some set X and taking its values in the field \mathbf{R} (or \mathbf{C}); its idempotent analog is a map $X \rightarrow S$, where X is some set and $S = \mathbf{R}_{\min}, \mathbf{R}_{\max}$, or another idempotent semiring. Let us show that redefinition of basic constructions of traditional calculus in terms of idempotent mathematics can yield interesting and nontrivial results (see, e.g., [32, 37–43, 45, 46, 48–53, 71, 72], for details, additional examples and generalizations).

Example 6. Integration and measures. To define an idempotent analog of the Riemann integral, consider a Riemann sum for a function $\varphi(x)$, $x \in X = [a, b]$, and substitute semiring operations \oplus and \odot for operations $+$ and \cdot (usual addition and multiplication) in its expression (for the sake of being definite, consider the semiring \mathbf{R}_{\max}):

$$\sum_{i=1}^N \varphi(x_i) \cdot \Delta_i \quad \mapsto \quad \bigoplus_{i=1}^N \varphi(x_i) \odot \Delta_i = \max_{i=1, \dots, N} (\varphi(x_i) + \Delta_i),$$

where $a = x_0 < x_1 < \dots < x_N = b$, $\Delta_i = x_i - x_{i-1}$, $i = 1, \dots, N$. As $\max_i \Delta_i \rightarrow 0$, the integral sum tends to

$$\int_X^{\oplus} \varphi(x) dx = \sup_{x \in X} \varphi(x)$$

for any function $\varphi: X \rightarrow \mathbf{R}_{\max}$ that is bounded. In general, for any set X the set function

$$m_\varphi(B) = \sup_{x \in B} \varphi(x), \quad B \subset X,$$

is called an \mathbf{R}_{\max} -measure on X ; since $m_\varphi(\bigcup_\alpha B_\alpha) = \sup_\alpha m_\varphi(B_\alpha)$, this measure is completely additive. An idempotent integral with respect to this measure is defined as

$$\int_X^\oplus \psi(x) dm_\varphi = \int_X^\oplus \psi(x) \odot \varphi(x) dx = \sup_{x \in X} (\psi(x) + \varphi(x)).$$

Using the standard partial order it is possible to generalize these definitions for the case of arbitrary idempotent semirings.

Example 7. Fourier–Legendre transform. Consider the topological group $G = \mathbf{R}^n$. The usual Fourier–Laplace transform is defined as

$$\varphi(x) \mapsto \tilde{\varphi}(\xi) = \int_G e^{i\xi \cdot x} \varphi(x) dx,$$

where $\exp(i\xi \cdot x)$ is a *character* of the group G , i.e., a solution of the following functional equation:

$$f(x + y) = f(x)f(y).$$

The idempotent analog of this equation is

$$f(x + y) = f(x) \odot f(y) = f(x) + f(y).$$

Hence natural “idempotent characters” of the group G are linear functions of the form $x \mapsto \xi \cdot x = \xi_1 x_1 + \dots + \xi_n x_n$. Thus the Fourier–Laplace transform turns into

$$\varphi(x) \mapsto \tilde{\varphi}(\xi) = \int_G^\oplus \xi \cdot x \odot \varphi(x) dx = \sup_{x \in G} (\xi \cdot x + \varphi(x)).$$

This is the well-known Legendre (or Fenchel) transform. Examples related to an important version of matrix algebra are discussed in section 4 above.

These examples suggest the following formulation of the idempotent correspondence principle [39, 40]:

There exists a heuristic correspondence between interesting, useful, and important constructions and results over

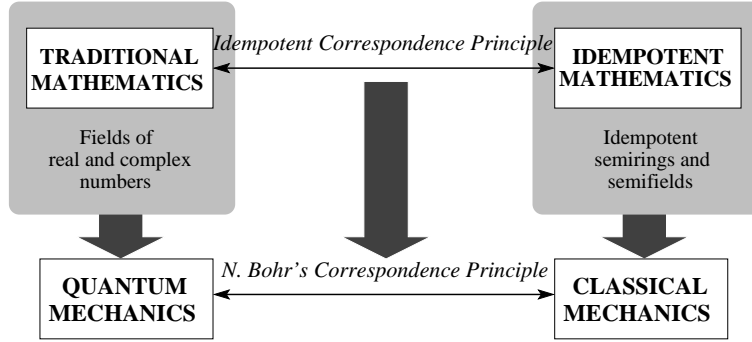


FIGURE 1. Relations between idempotent and traditional mathematics.

the field of real (or complex) numbers and similar constructions and results over idempotent semirings in the spirit of N. Bohr's correspondence principle in quantum mechanics.

So idempotent mathematics can be treated as a “classical shadow (or counterpart)” of the traditional Mathematics over fields.

A systematic application of this correspondence principle leads to a variety of theoretical and applied results, see, e.g. [37–43, 45, 46, 55, 71, 72]. Relations between idempotent and traditional mathematics are presented in Fig 1. Relations to quantum physics are discussed in detail, e.g., in [38].

7. THE SUPERPOSITION PRINCIPLE AND PARALLEL COMPUTING

In quantum mechanics the superposition principle means that the Schrödinger equation (which is basic for the theory) is linear. Similarly in idempotent mathematics the idempotent superposition principle means that some important and basic problems and equations (e.g., optimization problems, the Bellman equation and its versions and generalizations, the Hamilton–Jacobi equation) nonlinear in the usual sense can be treated as linear over appropriate idempotent semirings (in a general form this superposition principle was formulated by V. P. Maslov), see [48–53].

Linearity of the Hamilton-Jacobi equation over \mathbf{R}_{\min} (and \mathbf{R}_{\max}) can be deduced from the usual linearity (over \mathbf{C}) of the corresponding Schrödinger equation by means of the dequantization procedure described above (in Section 4). In this case the parameter h of this dequantization coincides with $i\hbar$, where \hbar is the Planck constant; so in this case \hbar must take imaginary values (because $h > 0$; see [38] for details). Of course, this is closely related to variational principles of mechanics.

The situation is similar for the differential Bellman equation (see, e.g., [32]) and the discrete version of the Bellman equations, see section 4 above.

It is well known that linear problems and equations are especially convenient for parallelization, see, e.g., [73]. Standard methods (including the so-called block methods) constructed in the framework of the traditional mathematics can be extended to universal algorithms over semirings (the correspondence principle!). For example, formula (1) discussed in section 4.3 leads to a simple block method for parallelization of the closure operations. Other standard methods of linear algebra [73] can be used in a similar way.

8. THE CORRESPONDENCE PRINCIPLE FOR COMPUTATIONS

Of course, the idempotent correspondence principle is valid for algorithms as well as for their software and hardware implementations [39, 40, 42, 43]. Thus:

If we have an important and interesting numerical algorithm, then there is a good chance that its semiring analogs are important and interesting as well.

In particular, according to the superposition principle, analogs of linear algebra algorithms are especially important. Note that numerical algorithms for standard infinite-dimensional linear problems over idempotent semirings (i.e., for problems related to idempotent integration, integral operators and transformations, the Hamilton-Jacobi and generalized Bellman equations) deal with the corresponding finite-dimensional (or finite) “linear approximations”. Nonlinear algorithms

often can be approximated by linear ones. Thus the idempotent linear algebra is a basis for the idempotent numerical analysis.

Moreover, it is well-known that linear algebra algorithms easily lend themselves to parallel computation; their idempotent analogs admit parallelization as well. Thus we obtain a systematic way of applying parallel computing to optimization problems.

Basic algorithms of linear algebra (such as inner product of two vectors, matrix addition and multiplication, etc.) often do not depend on concrete semirings, as well as on the nature of domains containing the elements of vectors and matrices. Algorithms to construct the closure $A^* = \mathbf{1} \oplus A \oplus A^2 \oplus \cdots \oplus A^n \oplus \cdots = \bigoplus_{n=1}^{\infty} A^n$ of an idempotent matrix A can be derived from standard methods for calculating $(\mathbf{1} - A)^{-1}$. For the Gauss–Jordan elimination method (via LU-decomposition) this trick was used in [66], and the corresponding algorithm is universal and can be applied both to the Bellman equation and to computing the inverse of a real (or complex) matrix $(\mathbf{1} - A)$. Computation of A^{-1} can be derived from this universal algorithm with some obvious cosmetic transformations.

Thus it seems reasonable to develop universal algorithms that can deal equally well with initial data of different domains sharing the same basic structure [39, 40, 43].

9. THE CORRESPONDENCE PRINCIPLE FOR HARDWARE DESIGN

A systematic application of the correspondence principle to computer calculations leads to a unifying approach to software and hardware design.

The most important and standard numerical algorithms have many hardware realizations in the form of technical devices or special processors. *These devices often can be used as prototypes for new hardware units generated by substitution of the usual arithmetic operations for its semiring analogs and by addition tools for performing neutral elements $\mathbf{0}$ and $\mathbf{1}$* (the latter usually is not difficult). Of course, the case of numerical semirings consisting of real numbers (maybe except neutral elements) and semirings of numerical intervals is the most simple and

natural [38–43, 45, 46, 69]. Note that for semifields (including \mathbf{R}_{\max} and \mathbf{R}_{\min}) the operation of division is also defined.

Good and efficient technical ideas and decisions can be transposed from prototypes into new hardware units. Thus the correspondence principle generated a regular heuristic method for hardware design. Note that to get a patent it is necessary to present the so-called ‘invention formula’, that is to indicate a prototype for the suggested device and the difference between these devices.

Consider (as a typical example) the most popular and important algorithm of computing the scalar product of two vectors:

$$(12) \quad (x, y) = x_1y_1 + x_2y_2 + \cdots + x_ny_n.$$

The universal version of (12) for any semiring A is obvious:

$$(13) \quad (x, y) = (x_1 \odot y_1) \oplus (x_2 \odot y_2) \oplus \cdots \oplus (x_n \odot y_n).$$

In the case $A = \mathbf{R}_{\max}$ this formula turns into the following one:

$$(14) \quad (x, y) = \max\{x_1 + y_1, x_2 + y_2, \cdots, x_n + y_n\}.$$

This calculation is standard for many optimization algorithms, so it is useful to construct a hardware unit for computing (14). There are many different devices (and patents) for computing (12) and every such device can be used as a prototype to construct a new device for computing (14) and even (13). Many processors for matrix multiplication and for other algorithms of linear algebra are based on computing scalar products and on the corresponding “elementary” devices respectively, etc.

There are some methods to make these new devices more universal than their prototypes. There is a modest collection of possible operations for standard numerical semirings: max, min, and the usual arithmetic operations. So, it is easy to construct programmable hardware processors with variable basic operations. Using modern technologies it is possible to construct cheap special-purpose multi-processor chips implementing examined algorithms. The so-called systolic processors are especially convenient for this purpose. A systolic array is a ‘homogeneous’ computing medium consisting of elementary processors, where the general scheme and processor connections are simple and regular.

Every elementary processor pumps data in and out performing elementary operations in a such way that the corresponding data flow is kept up in the computing medium; there is an analogy with the blood circulation and this is a reason for the term “systolic”, see e.g., [39, 40, 43, 52, 65–67].

Concrete systolic processors for the general algebraic path problem are presented in [65–67]. In particular, there is a systolic array of $n(n + 1)$ elementary processors which performs computations of the Gauss–Jordan elimination algorithm and can solve the algebraic path problem within $5n - 2$ time steps. Of course, hardware implementations for important and popular basic algorithms increase the speed of data processing.

The so-called GPGPU (General-Purpose computing on Graphics Processing Units) technique is another important field for applications of the correspondence principle. The thing is that graphic processing units (hidden in modern laptop and desktop computers) are potentially powerful processors for solving numerical problems. The recent tremendous progress in graphical processing hardware and software leads to new “open” programmable parallel computational devices (special processors), see, e.g., [78–80]. These devices are going to be standard for coming PC (personal computers) generations. Initially used for graphical processing only (at that time they were called GPU), today they are used for various fields, including audio and video processing, computer simulation, and encryption. But this list can be considerably enlarged following the correspondence principle: the basic operations would be used as parameters. Using the technique described in this paper (see also our references), standard linear algebra algorithms can be used for solving different problems in different areas. In fact, the hardware supports all operations needed for the most important idempotent semirings: plus, times, min, max. The most popular linear algebra packages [ATLAS (Automatically Tuned Linear Algebra Software), LAPACK, PLASMA (Parallel Linear Algebra for Scalable Multicore Architectures)] can already use GPGPU, see [81–83]. We propose to make these tools more powerful by using parameterized algorithms.

Linear algebra over the most important numerical semirings generates solutions for many concrete problems in different areas, see, e.g., sections 4.4 and 4.5 above and references indicated in these sections.

Note that to be consistent with operations we have to redefine zero (0) and unit (1) elements (see above); comparison operations must be also redefined as it is described in section 4.1 “Basic definitions.” Once the operations are redefined, then the most of basic linear algebra algorithms, including back and forward substitution, Gauss elimination method, Jordan elimination method and others could be rewritten for new domains and data structures. Combined with the power of the new parallel hardware this approach could change PC from entertainment devices to power full instruments.

10. THE CORRESPONDENCE PRINCIPLE FOR SOFTWARE DESIGN

Software implementations for universal semiring algorithms are not as efficient as hardware ones (with respect to the computation speed) but they are much more flexible. Program modules can deal with abstract (and variable) operations and data types. Concrete values for these operations and data types can be defined by the corresponding input data. In this case concrete operations and data types are generated by means of additional program modules. For programs written in this manner it is convenient to use special techniques of the so-called object oriented (and functional) design, see, e.g., [47, 62, 70]. Fortunately, powerful tools supporting the object-oriented software design have recently appeared including compilers for real and convenient programming languages (e.g. C^{++} and Java) and modern computer algebra systems.

Recently, this type of programming technique has been dubbed generic programming (see, e.g., [6, 62]). To help automate the generic programming, the so-called Standard Template Library (STL) was developed in the framework of C^{++} [62, 70]. However, high-level tools, such as STL, possess both obvious advantages and some disadvantages and must be used with caution.

It seems that it is natural to obtain an implementation of the correspondence principle approach to scientific calculations in the form of a

powerful software system based on a collection of universal algorithms. This approach ensures a working time reduction for programmers and users because of the software unification. The arbitrary necessary accuracy and safety of numeric calculations can be ensured as well.

The system has to contain several levels (including programmer and user levels) and many modules.

Roughly speaking, it must be divided into three parts. The first part contains modules that implement domain modules (finite representations of basic mathematical objects). The second part implements universal (invariant) calculation methods. The third part contains modules implementing model dependent algorithms. These modules may be used in user programs written in C^{++} , Java, Maple, Matlab etc.

The system has to contain the following modules:

- Domain modules:
 - infinite precision integers;
 - rational numbers;
 - finite precision rational numbers (see [44]);
 - finite precision complex rational numbers;
 - fixed- and floating-slash rational numbers;
 - complex rational numbers;
 - arbitrary precision floating-point real numbers;
 - arbitrary precision complex numbers;
 - p -adic numbers;
 - interval numbers;
 - ring of polynomials over different rings;
 - idempotent semirings;
 - interval idempotent semirings;
 - and others.
- Algorithms:
 - linear algebra;
 - numerical integration;
 - roots of polynomials;
 - spline interpolations and approximations;
 - rational and polynomial interpolations and approximations;

- special functions calculation;
- differential equations;
- optimization and optimal control;
- idempotent functional analysis;
- and others.

This software system may be especially useful for designers of algorithms, software engineers, students and mathematicians.

Note that there are some software systems oriented to calculations with idempotent semirings like \mathbf{R}_{\max} ; see, e.g., [64]. However these systems do not support universal algorithms.

11. INTERVAL ANALYSIS IN IDEMPOTENT MATHEMATICS

Traditional interval analysis is a nontrivial and popular mathematical area, see, e.g., [3, 20, 34, 54, 56, 59]. An “idempotent” version of interval analysis (and moreover interval analysis over positive semirings) appeared in [45, 46, 69]. Later appeared rather many publications on the subject, see, e.g., [12, 20, 27, 57, 58, 77]. Interval analysis over the positive semiring \mathbf{R}_+ was discussed in [8].

Let a set S be partially ordered by a relation \preceq . A *closed interval* in S is a subset of the form $\mathbf{x} = [\underline{\mathbf{x}}, \overline{\mathbf{x}}] = \{x \in S \mid \underline{\mathbf{x}} \preceq x \preceq \overline{\mathbf{x}}\}$, where the elements $\underline{\mathbf{x}} \preceq \overline{\mathbf{x}}$ are called *lower* and *upper bounds* of the interval \mathbf{x} . The order \preceq induces a partial ordering on the set of all closed intervals in S : $\mathbf{x} \preceq \mathbf{y}$ iff $\underline{\mathbf{x}} \preceq \underline{\mathbf{y}}$ and $\overline{\mathbf{x}} \preceq \overline{\mathbf{y}}$.

A *weak interval extension* $I(S)$ of a positive semiring S is the set of all closed intervals in S endowed with operations \oplus and \odot defined as $\mathbf{x} \oplus \mathbf{y} = [\underline{\mathbf{x}} \oplus \underline{\mathbf{y}}, \overline{\mathbf{x}} \oplus \overline{\mathbf{y}}]$, $\mathbf{x} \odot \mathbf{y} = [\underline{\mathbf{x}} \odot \underline{\mathbf{y}}, \overline{\mathbf{x}} \odot \overline{\mathbf{y}}]$ and a partial order induced by the order in S . The closure operation in $I(S)$ is defined by $\mathbf{x}^* = [\underline{\mathbf{x}}^*, \overline{\mathbf{x}}^*]$. There are some other interval extensions (including the so-called strong interval extension [46]) but the weak extension is more convenient.

The extension $I(S)$ is positive; $I(S)$ is idempotent if S is an idempotent semiring. A universal algorithm over S can be applied to $I(S)$ and we shall get an interval version of the initial algorithm. Usually both the versions have the same complexity. For the discrete stationary Bellman equation and the corresponding optimization problems on graphs

interval analysis was examined in [45, 46] in details. Other problems of idempotent linear algebra were examined in [12, 27, 57, 58, 77].

Idempotent mathematics appears to be remarkably simpler than its traditional analog. For example, in traditional interval arithmetic, multiplication of intervals is not distributive with respect to addition of intervals, whereas in idempotent interval arithmetic this distributivity is preserved. Moreover, in traditional interval analysis the set of all square interval matrices of a given order does not form even a semigroup with respect to matrix multiplication: this operation is not associative since distributivity is lost in the traditional interval arithmetic. On the contrary, in the idempotent (and positive) case associativity is preserved. Finally, in traditional interval analysis some problems of linear algebra, such as solution of a linear system of interval equations, can be very difficult (generally speaking, they are *NP*-hard, see [14, 20, 33, 34] and references therein). It was noticed in [45, 46] that in the idempotent case solving an interval linear system requires a polynomial number of operations (similarly to the usual Gauss elimination algorithm). Two properties that make the idempotent interval arithmetic so simple are monotonicity of arithmetic operations and positivity of all elements of an idempotent semiring.

Usually interval estimates in idempotent mathematics are exact. In the traditional theory such estimates tend to be overly pessimistic.

ACKNOWLEDGMENTS

This work is partially supported by the RFBR grant 08-01-00601. The authors are grateful to A. G. Kushner for his kind help.

REFERENCES

- [1] A. V. Aho and J. D. Ullman, *The theory of parsing, translation and compiling, Vol. 2: Compiling*, Prentice-Hall, Englewood Cliffs, N. J., 1973.
- [2] A. V. Aho, J. E. Hopcroft and J. D. Ullman, *The design and analysis of computer algorithms*, Addison-Wesley Publ. Co., Reading (Massachusetts) et al., 1976.
- [3] G. Alefeld and J. Herzberger, *Introduction to Interval Computations*, Academic Press, New York, 1983.

- [4] S. M. Avdoshin, V. V. Belov, V. P. Maslov, and A. M. Chebotarev, *Design of computational media: mathematical aspects*. – In: [53], p. 9-145.
- [5] F. Baccelli, G. Cohen, G. J. Olsder, and J.-P. Quadrat, *Synchronization and Linearity: An Algebra for Discrete Event Systems*, John Wiley & Sons Publishers, New York e.a., 1992.
- [6] R. Backhouse, P. Jansson, J. Jeuring, L. Meertens, *Generic Programming – An Introduction*, Lect. Notes Comput. Sci. **1608** (1999), 28–115.
- [7] R. C. Backhouse, and B. A. Carré, *Regular Algebra Applied to Path-finding Problems*, J. Inst. Math. Appl **15** (1975), 161–186.
- [8] W. Barth and E. Nuding, *Optimale Lösung von Intervallgleichungssystemen, Computing*, **12** (1974), 117–125.
- [9] P. Butkovič, K. Zimmermann, *A strongly polynomial algorithm for solving two-sided linear systems in max-algebra*, Discrete Applied Mathematics, **154** (2006), 437–446.
- [10] B. A. Carré, *An algebra for network routing problems*, J. Inst. Appl. **7** (1971), 273-294.
- [11] B. A. Carré, *Graphs and networks*, The Clarendon Press/Oxford University Press, Oxford, 1979.
- [12] K. Cechlárová and R. A. Cuninghame-Green, *Interval systems of max-separable linear equations*, Linear Algebra and its Applications **340** (2002), 215–224.
- [13] G. Cohen, S. Gaubert, and J. P. Quadrat, *Max-plus algebra and system theory: where we are and where to go now*, Annual Reviews in Control **23** (1999), 207–219.
- [14] G. E. Coxson, *Computing exact bounds on elements of an inverse interval matrix is NP-hard*, Reliable Computing, **5** (1999), pp. 137–142.
- [15] R. A. Cuninghame-Green, *Minimax algebra*, Springer Lect. Notes in Economics and Mathematical Systems **166**, Berlin et al., 1979.
- [16] R. A. Cuninghame-Green, *Minimax algebra and applications*, Advances in Imaging and Electron Physics **90** (1995), 1–121. (Academic Press, New York).
- [17] R. A. Cuninghame-Green, *Minimax algebra and its applications*, Fuzzy Sets and Systems **41**, (1991), 251–267.
- [18] R. A. Cuninghame-Green and P. Butkovic, *The equation $a \otimes x = b \otimes y$ over $(\max, +)$* . Theoretical Computer Science, **293** (2003), 3–12.
- [19] P. Del Moral, *A survey of Maslov optimization theory: optimality versus randomness*. – In: V. N. Kolokoltsov and V. P. Maslov, *Idempotent Analysis and Applications*, Kluwer Acad. Publ., Dordrecht, 1997, p. 243–302 (Appendix).
- [20] M. Fiedler, J. Nedoma, J. Ramik, J. Rohn, K. Zimmermann, *Linear Optimization Problems with Inexact Data*, Springer, New York, 2006.

- [21] K. Glazek, *A guide to the literature on semirings and their applications in mathematics and information sciences: with complete bibliography*, Kluwer Acad. Publ., Dordrecht e.a., 2002.
- [22] J. S. Golan, *Semirings and their applications*, Kluwer Acad. Publ., Dordrecht, 1999.
- [23] M. Gondran, *Path algebra and algorithms*. – In: *Combinatorial programming: methods and applications* (B. Roy, ed.), NATO Adv. Study Inst. Ser., Ser. C., **19**, 1975, 137–148.
- [24] M. Gondran and M. Minoux, *Graphes et algorithmes*, Editions Eyrolles, Paris, 1979, 1988.
- [25] M. Gondran and M. Minoux, *Graphes, diïodes et semi-anneaux*, Editions TEC&DOC, Paris e.a., 2001.
- [26] J. Gunawardena (Ed.), *Idempotency*, Publ. of the Newton Institute, Vol. **11**, Cambridge University Press, Cambridge, 1998.
- [27] L. Hardouin, B. Cottenceau, M. Lhommeau, E. Le Corrond, *Interval systems over idempotent semirings*, Linear Algebra and its Applications, **431** (2009), 855–862.
- [28] I. Itenberg, G. Mikhalkin, E. Shustin, *Tropical Algebraic Geometry*, Oberwolfach Seminars, Vol. **35**, Birkhäuser, Basel e.a., 2007.
- [29] S. C. Kleene, *Representation of events in nerve sets and finite automata*. – In: J. McCarthy and C. Shannon (Eds), Automata Studies, Princeton University Press, Princeton, 1956, pp. 3–40.
- [30] E. P. Klement and E. Pap (Eds.), *Mathematics of Fuzzy Systems*, 25th Linz Seminar on Fuzzy Set Theory, Linz, Austria, Feb. 3–7, 2004. Abstracts. J. Kepler Univ., Linz, 2004.
- [31] V. N. Kolokoltsov, *Idempotency structures in optimization*, Journal Math. Sci. **104** (2001), no. 1, 847–880.
- [32] V. Kolokoltsov and V. Maslov, *Idempotent analysis and applications*, Kluwer Acad. Publ., 1997.
- [33] V. Kreinovich, A. V. Lakeyev and S. I. Noskov, *Optimal solution of interval linear systems is intractable (NP-hard)*, Interval Computations **1** (1993), 6–14.
- [34] V. Kreinovich, A. V. Lakeyev, J. Rohn, and P. Kahl, *Computational Complexity and Feasibility of Data Processing and Interval Computations*, Kluwer Academic Publishers, Dordrecht, 1998.
- [35] H. T. Kung, *Two-level pipelined systolic arrays for matrix multiplication, polynomial evaluation and discrete Fourier transformation*. – In: *Dynamical Systems and Cellular Automata* (J. Demongeot et al., Eds), Academic Press, New York et al., 1985, 321–330.

- [36] D. J. Lehmann, *Algebraic Structures for Transitive Closure*, Theor. Comput. Sci., **4** (1977), 59–76.
- [37] G. L. Litvinov, *Dequantization of mathematics, idempotent semirings and fuzzy sets*. — In [30], p. 113–117.
- [38] G. L. Litvinov, *The Maslov dequantization, idempotent and tropical mathematics: a brief introduction.*, Journal of Mathematical Sciences, **140**, #3 (2007), 426–444. See also E-print arXiv:math.GM/0507014 (<http://arXiv.org>).
- [39] G. L. Litvinov and V. P. Maslov, *Correspondence principle for idempotent calculus and some computer applications*, (IHES/M/95/33), Institut des Hautes Etudes Scientifiques, Bures-sur-Yvette, 1995. Also arXiv:math.GM/0101021.
- [40] G. L. Litvinov and V. P. Maslov, *The correspondence principle for idempotent calculus and some computer applications*. — In [26], p. 420–443.
- [41] G. L. Litvinov and V. P. Maslov (Eds.), *Idempotent mathematics and mathematical physics*, Contemporary Mathematics, Vol. 377, AMS, Providence, RI, 2005.
- [42] G. L. Litvinov and E. V. Maslova, *Universal numerical algorithms and their software implementation*, Programming and Computer Software **26** (2000), no. 5, 275–280. Also arXiv:math.SC/0102114.
- [43] G. L. Litvinov, V. P. Maslov, and A. Ya. Rodionov, *A unifying approach to software and hardware design for scientific calculations and idempotent mathematics*, International Sophus Lie Centre, Moscow 2000. Also arXiv:math.SC/0101069.
- [44] G. L. Litvinov, A. Ya. Rodionov, and A. V. Tchourkin, *Approximate rational arithmetics and arbitrary precision computations for universal algorithms*, International Journal of Pure and Applied Mathematics, **45**, No.2, (2008), 193–204. See also E-print math.NA/0101152 (<http://ArXiv.org>).
- [45] G. L. Litvinov and A. N. Sobolevskii, *Exact interval solutions of the discrete Bellman equation and polynomial complexity of problems in interval idempotent linear algebra*, Doklady Mathematics **62** (2000), no. 2, 199–201. Also arXiv:math.LA/0101041.
- [46] G. L. Litvinov and A. N. Sobolevskii, *Idempotent interval analysis and optimization problems*, Reliable Computing **7** (2001), no. 5, 353–377. Also arXiv:math.SC/0101080.
- [47] M. Lorenz, *Object oriented software development: a practical guide*, Prentice Hall Books, Englewood Cliffs, N.J., 1993.
- [48] V. P. Maslov, *New superposition principle for optimization problems*. — In: Seminaire sur les Equations aux Dérivées Partielles 1985/86, Centre Math. De l'Ecole Polytechnique, Palaiseau, 1986, exposé 24.

- [49] V. P. Maslov, *A new approach to generalized solutions of nonlinear systems*, Soviet Math. Dokl. **42** no.1, (1987), 29–33.
- [50] V. P. Maslov, *On a new superposition principle for optimization problems*, Uspekhi Mat. Nauk, [Russian Math. Surveys], **42**, no. 3 (1987), 39–48.
- [51] V. P. Maslov, *Méthodes opératoriellles*, Mir, Moscow, 1987.
- [52] V. P. Maslov et al., *Mathematics of semirings and its applications*, Technical report. Institute for New Technologies, Moscow 1991(in Russian).
- [53] V. P. Maslov and K. A. Volosov (Eds.), *Mathematical aspects of computer engineering*, MIR Publ., Moscow, 1988.
- [54] Yu. V. Matijasevich, *A posteriori version of interval analysis*. – In: M. Arató, I. Kátaí and L. Varga (eds.), *Topics in the Theoretical Basis and Applications of Computer Sciences. Proc. Fourth Hung. Computer Sci. Conf.*, Budapest: Acad. Kiado, 1986, 339–349.
- [55] G. Mikhalkin, *Tropical geometry and its applications*, Proceedings of the ICM, Madrid, Spain, vol. II, 2006, pp. 827–852. Also arXiv:math.AG/0601041v2.
- [56] R. E. Moore, *Methods and Applications of Interval Analysis*, SIAM, Philadelphia, 1979.
- [57] H. Myskova, *Interval systems of max-separable linear equations*, Linear Algebra and its Applications, **403** (2005), 263–272.
- [58] H. Myskova, *Control solvability of interval systems of max-separable linear equations*, Linear Algebra and its Applications, **416** (2006), 215–223.
- [59] A. Neumayer, *Interval Methods for Systems of Equations*, Cambridge University Press, Cambridge, 1990.
- [60] S. N. N. Pandit, *A new matrix calculus*, SIAM J. Appl. Math. **9** (1961), 632–639.
- [61] J. E. Pin, *Tropical semirings*. – In: [26], p. 50–60.
- [62] I. Pohl, *Object-Oriented Programming Using C^{++}* . Reading: Addison-Wesley, 1997, 2nd ed.
- [63] J.-P. Quadrat, *Théorèmes asymptotiques en programmation dynamique*, Comptes Rendus Acad. Sci., Paris **311**, (1990), 745–748.
- [64] J.-P. Quadrat and Max-Plus working group, *Maxplus Algebra Software*. <http://scilab.org/contrib>. <http://maxplus.org>. 2007.
- [65] Y. Robert and D. Tristram, *An orthogonal systolic array for the algebraic path problem*, Computing **39**, (1987), 187–199.
- [66] G. Rote, *A Systolic Array Algorithm for the Algebraic Path Problem (Shortest Paths; Matrix Inversion)*, Computing **34** (1985), 191–219.
- [67] S. G. Sedukhin, *Design and analysis of systolic algorithms for the algebraic path problem*, Computers and Artificial Intelligence **11**, #3 (1992), 269–292.
- [68] I. Simon, *Recognizable sets with multiplicities in the tropical semiring*. Lecture Notes in Computer Science **324** (1988), 107–120.

- [69] A. N. Sobolevskii, *Interval arithmetic and linear algebra over idempotent semirings*, Doklady Akademii Nauk **369** (1999), 747–749 (in Russian).
- [70] A. Stepanov and M. Lee, *The Standard Template Library*, Palo Alto: Hewlett-Packard, 1994.
- [71] O. Viro, *Dequantization of real algebraic geometry on a logarithmic paper*. — In: 3rd European Congress of Mathematics, Barcelona, 2000. Also arXiv:math/0005163.
- [72] O. Viro, *From the sixteenth Hilbert problem to tropical geometry*, Japan. J. Math. **3** (2008), 1–30.
- [73] V. V. Voevodin, *Mathematical foundation of parallel computings*, World Scientific Publ. Co., Singapore, 1992, 343 pp.
- [74] N. N. Vorobjev, *The extremal matrix algebra*, Soviet Math. Dokl. **4** (1963), 1220–1223.
- [75] N. N. Vorobjev, *Extremal algebra of positive matrices*, Elektronische Informationsverarbeitung und Kybernetik **3** (1967), 39–57. (in Russian)
- [76] N. N. Vorobjev, *Extremal algebra of nonnegative matrices*, Elektronische Informationsverarbeitung und Kybernetik **6** (1970), 302–312. (In Russian).
- [77] K. Zimmermann, *Interval linear systems and optimization problems over max-algebras*. — In: [20], p. 165–193.
- [78] 2009 IEEE International Symposium on Parallel & Distributed Processing. Rome, Italy, May 23–May 29. ISBN: 978-1-4244-3751-1.
- [79] D. Blithe, *Rise of the graphics processor*, Proc. of the IEEE, **96**, no. 5 (2008), 761–778.
- [80] J. D. Owens et al., *GPU computing*, Ibid, 879–899.
- [81] ATLAS: <http://math-atlas.sourceforge.net/> .
- [82] LAPACK: <http://www.netlib.org/lapack/> .
- [83] PLASMA: <http://icl.cs.utk.edu/plasma/> .